Problem 1. Felipe and the Sequence

Source file name:	Sequence.c, Sequence.cpp, Sequence.java, Sequence.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2017 - Contest 03 RPC 2017`

On February 19, 2017, Red Matemática proposed the following mathematical challenge on its twitter account (@redmatematicant): "Felipe, how many terms of the next sequence of numbers must be added to make the result equal to 200?"

$$\frac{1}{\sqrt{1}+\sqrt{2}}+\frac{1}{\sqrt{2}+\sqrt{3}}+\frac{1}{\sqrt{3}+\sqrt{4}}+\frac{1}{\sqrt{4}+\sqrt{5}}+\cdots=200$$

Using this interesting puzzle as our starting point, the problem you are asked to solve now is: Given a positive integer S ($1 \leq S \leq 10^9$) representing the result obtained for the sum of the terms in the sequence, find out the number n that represents the total number of terms in the sequence to sum up.

$$\frac{1}{\sqrt{1}+\sqrt{2}}+\frac{1}{\sqrt{2}+\sqrt{3}}+\frac{1}{\sqrt{3}+\sqrt{4}}+\frac{1}{\sqrt{4}+\sqrt{5}}+\cdots+\frac{1}{\sqrt{n}+\sqrt{n+1}}=S$$

Input

Input begins with an integer t ($1 \leq t \leq 5 * 10^5$), the number of test cases, followed by t lines, each containing an integer S ($1 \leq S \leq 10^9$).

Output

For each test case, your program must print one positive integer denoting the number n that represents the total number of terms in the sequence to sum up.

Example

Input	Output
3 200 725 1333	40400 527075 1779555

Problem 2. Triangular Numbers

Source file name:	Triangular.c, Triangular.cpp, Triangular.java, Triangular.py
Input:	Standard
Output:	Standard
Author(s):	Hugo Humberto Morales Peña - UTP Colombia
Source:	UTP Open 2013

Triangular Numbers are positive integer numbers such that they represent an amount of "dots" with which you can form a compact equilateral triangle of dots.

The first five triangular numbers are:

1	3	6	10	15
*	* * *	* * * * * *	* * * * * * * * * *	* * * * * * * * * * * * * * *

For this problem, you must create a program that determines if a given number n is triangular or not.

Input

Input may contain several test cases. Each test case is given in a line of its own, and contains an integer n ($1 \leq n \leq 16 \times 10^{18}$). Input ends with a test case in which n is zero, and it must not be processed.

Output

For each test case given in the input, your program must print **YES** or **NO**, indicating whether n is a triangular number or not. There must be a single line of output for each test case.

Example

Input	Output
1 15 16 101 15999999994386249876 0	YES YES NO NO YES

Problem 3. Johann's Function

Source file name:	Johann.c, Johann.cpp, Johann.java, Johann.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`ICPC Centroamérica 2020 - Contest 01 RPC 2021`

Typically in a first-year programming course in an engineering or computer science academic program, students are taught to build functions that make use of the doubly nested loop structure such as the following:

```
unsigned long long JohannsFunction(int n)
{
    int i, j;
    unsigned long long result = 0;

    for(i = 1; i <= n; i++)
    {
        for(j = 1; j <= i; j++)
        {
            result += j;
        }
    }

    return result;
}
```

The running-time function of the previous algorithm belongs to $O(n^2)$, with a value of $n = 10^6$ the total number of operations performed by the algorithm, in order to give the result would require 10^{12} steps. As a competitor of the different phases of the ICPC, you know that a solution that performs that amount of steps, will obtain a verdict of **Time Limit Exceeded** in competition, for this reason you are asked to propose a solution that has a running time as close to $O(1)$ as possible, regardless of the size of n, in which you must make use of all your experience as a competitor of ICPC!

Input

The input begins with a positive integer t ($1 \leq t \leq 10^6$), which represents the total number of test cases. Then t lines are presented, each one containing a positive integer n ($1 \leq n \leq 10^6$) for which the result of the **Johann's Function** must be calculated.

Output

The output must contain t lines, each containing a long positive integer as a result of the **Johann's Function**.

Example

Input	Output
7 1 10 100 1000 10000 100000 1000000	1 220 171700 167167000 166716670000 166671666700000 166667166667000000

Problem 4. Standard Deviation

Source file name:	Deviation.c, Deviation.cpp, Deviation.java, Deviation.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2016 - Contest 03 RPC 2016`

In mathematics, the standard deviation of a set of n integer numbers is defined as:

$$S = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}}$$

where $\overline{x}$ is the average of the set of n integer numbers for which the standard deviation is being calculated. That average is calculated as:

$$\overline{x} = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i$$

The task is to calculate, **in an efficient way**, the standard deviation of the first n odd positive integer numbers.

Input

There are several test cases in the input. Each test case consists of a single line containing a positive integer number n ($2 \leq n \leq 10^6$) which indicates the amount of consecutive odd numbers (starting from one) that should be considered when calculating the standard deviation. The last test case has a value of 0, for which you shouldn't generate any response.

Output

For each test case, you should print a single line containing a floating point number: the standard deviation of the first n odd positive numbers. The absolute error of your answer should not be greater than 10^{-6}.

Example

Input	Output
10	6.055301
100	58.022984
1000	577.638872
10000	5773.791360
100000	57735.315593
1000000	577350.557865
0	

Use fast I/O methods

Problem 5. Triangular Number Sum Test

Source file name:	Sumtest.c, Sumtest.cpp, Sumtest.java, Sumtest.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2016 - Contest 03 RPC 2016`

En 1638, Fermat propuso que cada número entero positivo es la suma de como máximo tres números triangulares, cuatro números cuadrados, cinco números pentagonales, y n números n-poligonales. Esto es lo que se conoce como el **Teorema de los números Poligonales de Fermat**.

Apoyados en el **Teorema de los Números Poligonales de Fermat**, en este reto se debe calcular la cantidad mínima de números triangulares que se deben sumar para obtener un número entero positivo n.

Input

La entrada del problema consiste de varios casos de prueba. Cada caso de prueba es presentado en una sola línea que contiene un número entero positivo n ($1 \leq n \leq 3 \cdot 10^6$). El final de los casos de prueba es dado por el fin de archivo (`End Of File - EOF`).

Output

Para cada caso de prueba, su programa debe imprimir un número entero positivo que denota la mínima cantidad de números triangulares, cuya suma es igual a n. Cada caso de prueba debe generar una sola línea en la salida.

Example

Input	Output
1	1
2	2
3	1
4	2
5	3
6	1
7	2
8	3
9	2
10	1
11	2
49	2

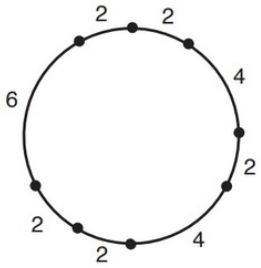
Use fast I/O methods

Problem 6. Triangles

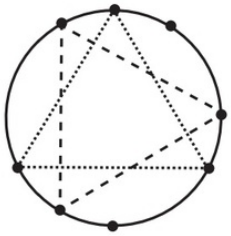
Source file name:	Triangles.c, Triangles.cpp, Triangles.java, Triangles.py
Input:	`Standard`
Output:	`Standard`
Source:	`Maratona de Programacao da SBC 2013 - Primeira Fase`

You will be given N points on a circle. You must write a program to determine how many distinct equilateral triangles can be constructed using the given points as vertices.

The figure below illustrates an example: (a) shows a set of points, determined by the lengths of the circular arcs that have adjacent points as extremes; and (b) shows the two triangles which can be built with these points.



(a)



(b)

Input

The input contains several test cases. The first line of a test case contains an integer N ($3 \leq N \leq 10^5$), the number of points given. The second line contains N integers X_i ($1 \leq X_i \leq 10^3$, for $1 \leq i \leq N$), representing the lengths of the circular arcs between two consecutive points in the circle: for $1 \leq i \leq (N-1)$, X_i represents the length of the arc between between points i and $i+1$; X_N represents the length of the arc between points N and 1.

Output

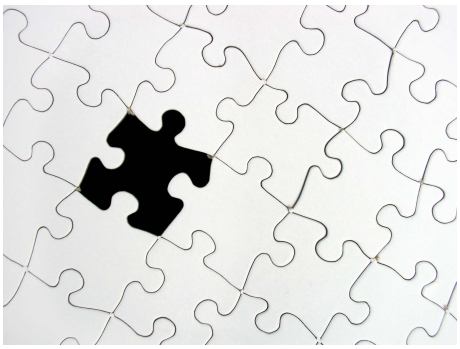
For each test case your program must output a single line, containing a single integer, the number of distinct equilateral triangles that can be constructed using the given points as vertices.

Example

Input	Output
8 4 2 4 2 2 6 2 2 8 2 2 2 2 2 4 4 6 8 6 4 4 2 2 2 2 2 8 1 1 1 1 1 1 1 1 9 1 1 1 1 1 1 1 1 1 6 3 4 2 1 5 3	2 1 1 0 3 1

Problem 7. Impossible

Source file name:	Impossible.c, Impossible.cpp, Impossible.java, Impossible.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Milton Jesús Vera Contreras - UFPS Colombia`
Source:	`Maratón de Programación UFPS 2022 - Contest 09 RPC 2022`



Two mathematicians and one engineer are auditing bank ATM transactions. Each transaction is identified with a unique ID number that is consecutive. They have the unordered transactions list, but one ID number is missing. Mathematicians say that it is not possible find the missing ID number. The engineer says that there are cases that can be solved. May you help them?

Input

Several test cases. Each test case starts with a number $10^3 \leq n \leq 10^6$, which corresponds to the number of transactions. Then follows a list of $n-1$ ID numbers x_i, $10^5 \leq x_i \leq 10^9$. The ID numbers are consecutive and unordered, there are not duplicate numbers and one number is missing. Each number is separated by a blank space.

Output

For each test case print one line with the ID number of transaction x_i that is missing or `IMPOSSIBLE` if the mathematicians are correct and "it is not possible find the missing ID number".

Example

Input	Output
10 123465 123460 123459 123457 123458 123463 123461 123464 123456	123462
11 789065 789060 789059 789057 789058 789062 789063 789061 789064 789056	IMPOSSIBLE

Note: Due to the amount of data, an example is used with a number n smaller.

Problem 8. How Many Triangular Numbers?

Source file name:	Howmany.c, Howmany.cpp, Howmany.java, Howmany.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UCO - Code Fortress 2 - 2014.`

Triangular Numbers are positive integer numbers such that they represent an amount of "dots" with which you can form a compact equilateral triangle of dots.

The first five triangular numbers are:

1	3	6	10	15
*	* *	* * *	* * * *	* * * * *

For this problem, given two long positive integers a and b, calculate how many triangular numbers exist in the closed interval $[a,\ b]$.

Input

The input consist of several test cases. Every test case is presented in a single line and contains two long positive integers a and b ($1 \leq a \leq b \leq 2 \cdot 10^{12}$). Input ends with a test case in which a and b are zero, and it must not be processed.

Output

For every test case, print a single line with the quantity of triangular numbers t_i that $a \leq t_i \leq b$.

Example

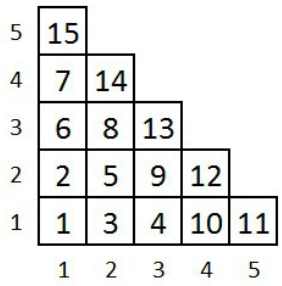
Input	Output
1 15 2 15 1 14 2 14 5 5 6 6 0 0	5 4 4 3 0 1

Problem 9. Dora the Explorer I

Source file name:	Dora01.c, Dora01.cpp, Dora01.java, Dora01.py
Input:	`Standar`
Output:	`Standar`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2012`

One day, there was this little ant called "Dora the Explorer", the ant arrived to a triangular board of d diagonals. the ant wanted to explore every boxes of the board, so she began to walking on this board starting on the diagonal containing only one box.

Dora the explorer began on the box (1, 1). In the first place she took one step forward to the box (1, 2), then she descended by the diagonal to reach the box (2, 1), after that, the ant took one step to the right to the box (3, 1), then she ascended by the respective diagonal passing trough the boxes (2, 2) and (1, 3). Each time the ant added a new diagonal to her trajectory either ascending or descending by the diagonal depending of the first box she reached in the respective diagonal.



For example, the ant in her first 15 steps made the following trajectory in the triangular board, where the number of each box or cell indicates the order in which it was visited:

The 10th step was in the box (4, 1), while the 15th step was in the box (1, 5). The objective consists of determining "Dora's the ant" position in triangular board given an step, assuming that the triangular board is big enough to admit any step.

Input

Each line of the input contains an integer n, which indicates the number of the step, where $(1 \leq n \leq 2 \times 10^{18})$. The input ends with a line containing 0.

Output

For each test case, print a line with two integer numbers $(x,\ y)$ indicating the values of the column and the row, respectively. between x and y there's only a white space.

Example

Input	Output
1	1 1
6	1 3
10	4 1
11	5 1
15	1 5
16	1 6
17	2 5
0	

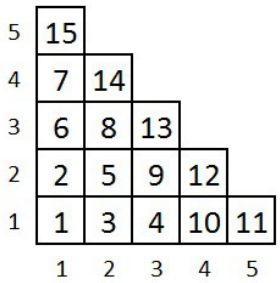
Problem 10. Dora the Explorer II

Source file name:	Dora02.c, Dora02.cpp, Dora02.java, Dora02.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`Maratón de Programación UFPS 2019 - Contest 06 RPC 2019`

One day, there was this little ant called "Dora the Explorer", the ant arrived to a triangular board of d diagonals. the ant wanted to explore every boxes of the board, so she began to walking on this board starting on the diagonal containing only one box.

Dora the explorer began on the box $(1,\ 1)$. In the first place she took one step forward to the box $(1,\ 2)$, then she descended by the diagonal to reach the box $(2,\ 1)$, after that, the ant took one step to the right to the box $(3,\ 1)$, then she ascended by the respective diagonal passing trough the boxes $(2,\ 2)$ and $(1,\ 3)$. Each time the ant added a new diagonal to her trajectory either ascending or descending by the diagonal depending of the first box she reached in the respective diagonal.

For example, the ant in her first 15 steps made the following trajectory in the triangular board, where the number of each box or cell indicates the order in which it was visited:



The 10th step was in the box $(4,\ 1)$, while the 15th step was in the box $(1,\ 5)$.

The objective is to determine the number of steps that Dora the Explorer must make in her trajectory to reach the box $(x,\ y)$ of the triangular board. You will suppose that to reach the box $(1,\ 1)$ is necessary one step, this box is the origin on the triangular board. Additionally, you will be assumed that the triangular board is big enough to admit the coordinates of any box.

Input

The input contains several test cases. Each line of the input contains two space-separate positive integer numbers $x\ y$ $(1 \leq x,\ y \leq 2 \times 10^9)$, denoting the coordinates of the box with the column and row values, respectively. The input ends with a line containing 0 0, this case must not be processed.

Output

For each test case, print a line with a positive integer number n indicating the number of steps that Dora the Explorer must make in her trajectory to reach the box.

Example

Input	Output
1 1 1 3 4 1 5 1 1 5 1 6 2 5 0 0	1 6 10 11 15 16 17

Use fast I/O methods

Problem 11. The Book Thief

Source file name:	Book.c, Book.cpp, Book.java, Book.py
Input:	Standard
Output:	Standard
Author(s):	Hugo Humberto Morales Peña - UTP Colombia
Source:	UTP Open 2015 - Contest 03 RPC 2015

On February 18, 2014, Red Matemática proposed the following mathematical challenge on their twitter account (@redmatematicant): "While Anita read: *The book thief* by Markus Zusak, She added all the page numbers starting from 1. When she finished the book, she got a sum equal to 9.000 but she realized that one page number was forgotten in the process. What is such number? and, how many pages does the book have?"

Using this interesting puzzle as our starting point, the problem you are asked to solve now is: Given a positive integer s ($1 \leq s \leq 10^8$) representing the result obtained by Anita, find out the number of the forgotten page and the total number of pages in the book.

Input

The input may contain several test cases. Each test case is presented on a single line, and contains one positive integer s. The input ends with a test case in which s is zero, and this case must not be processed.

Output

For each test case, your program must print two positive integers, separated by a space, denoting the number of the forgotten page and the total number pages in the book. Each valid test case must generate just one output line.

Example

Input	Output
1	2 2
2	1 2
3	3 3
4	2 3
5	1 3
6	4 4
9000	45 134
499977	523 1000
49999775	5225 10000
0	

Use faster I/O methods

Problem 12. Numeric Center (Small)

Source file name:	Center.c, Center.cpp, Center.java, Center.py
Input:	Standard
Output:	Standard
Author(s):	Hugo Humberto Morales Peña - UTP Colombia
Source:	UTP Open 2015 - Contest 03 RPC 2015

A numeric center is a number that separates in a consecutive and positive integer number list (starting at one) in two groups of consecutive and positive integer numbers, in which their sum is the same. The first numeric center is number 6, which takes the list $\{1, 2, 3, 4, 5, 6, 7, 8\}$ and produces two lists of consecutive and positive integer numbers in which their sum (in this case 15) is the same. Those lists are: $\{1, 2, 3, 4, 5\}$ and $\{7, 8\}$. The second numeric center is 35, that takes the list $\{1, 2, 3, 4, \ldots, 49\}$ and produces the following two lists: $\{1, 2, 3, 4, \ldots, 34\}$ and $\{36, 37, 38, 39, \ldots, 49\}$, the sum of each list is equal to 595.

The task consists in writing a program that calculates the total of numeric centers between 1 and n.

Input

The input consists of several test cases. There is only one line for each test case. This line contains a positive integer number n ($1 \leq n \leq 10^6$). The last test case is a value of n equal to zero, this test case should not be processed.

Output

For each test case you have to print in one line, the number of numeric centers between 1 and n.

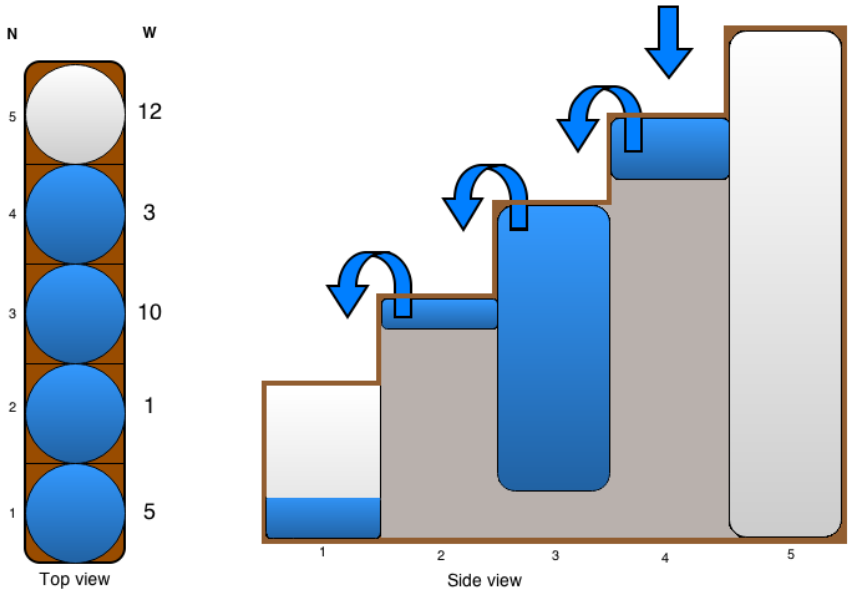
Example

Input	Output
1	0
7	0
8	1
48	1
49	2
50	2
0	

Problem 13. Tobby and Tanks II

Source file name:	Tanks02.c, Tanks02.cpp, Tanks02.java, Tanks02.py
Input:	Standard
Output:	Standard
Author(s):	Jhon Jimenez - UTP Colombia
Source:	UTP Open 2014 - Contest 10 RPC 2021

Tobby is a curious dog and he is always thinking about new and revolutionary ideas. Now he imagined N tanks placed in a row (staggered - See the figure). Every tank has a capacity of W liters of water. The cute and curious dog is asked himself, if you have K liters of water, which is the i-th tank (where i is the maximum possible), such that if we drop the water there, the water will reach the first tank. As you know Tobby is not a complicated dog, so he does not want completely fill the first tank, he will be happy if the first tank have at least 1 liter of water.



You are given n tanks with capacity w_i, and q queries, each query contains one integer k, the amount of water.

Input

For every test case, the first line contains two integers n ($1 \leq n \leq 10^5$) and q ($1 \leq q \leq 10^4$), the number of tanks and the number of queries respectively, the next line contains n integers w_i ($1 \leq w \leq 10^4$). The next q lines contains a single integer k ($1 \leq k \leq 10^9$), the amount of water. (Read until EOF).

Output

For every query you must print a single integer, the maximum i ($1 \leq i \leq n$)

Example

Input	Output
5 2 5 1 10 3 12 16 1	4 1

Problem 14. Subset Frequencies

Source file name:	Frequencies.c, Frequencies.cpp, Frequencies.java, Frequencies.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`Coding Cup 2018 - ITSUR México`

Se tienen dos conjuntos A y S de números enteros positivos, el conjunto A tiene cardinalidad n ($|A| = n$) y el conjunto S tiene cardinalidad k ($|S| = k$), recordar que la cardinalidad de un conjunto es el total de elementos distintos que este contiene.

La tarea a realizar consiste en lo siguiente, para cada uno de los s_i elementos del conjunto S se quiere averiguar cuantos subconjuntos de tamaño dos en el conjunto A tienen una suma de sus elementos igual a el. Para hacer esto aun más interesante, la salida debe cumplir el ordenamiento que se pide en la sección "**Output**"!

Input

La entrada del problema consiste de un único caso de prueba. La primera línea del caso de prueba contiene dos números enteros positivos n y k ($3 \leq n \leq 10^6$, $1 \leq k \leq 10^2$), que representan respectivamente las cardinalidades de los conjuntos A y S. Luego, la segunda línea del caso de prueba contiene los n números enteros positivos a_i del conjunto A ($1 \leq a_i \leq 10^8$, $1 \leq i \leq n$), obviamente se garantiza que los n elementos del conjunto A son diferentes. La tercera línea del caso de prueba contiene los k números enteros positivos s_j del conjunto S ($1 \leq s_j \leq 2 \times 10^8$, $1 \leq j \leq k$), de nuevo se garantiza que los k elementos del conjunto S son diferentes.

Output

Su programa debe imprimir k líneas, cada una de ellas conteniendo dos números enteros positivos s_i f_i ($1 \leq i \leq k$) los cuales representan el valor de la suma de los dos elementos del subconjunto de tamaño dos y la frecuencia de ocurrencia de dichos subconjuntos sobre el conjunto A.

Nota: Las k líneas de la salida del programa deben estar ordenadas de forma descendente con respecto a el valor f_i, en el caso de líneas que tenga el mismo valor de f_i, entonces se debe ordenar de forma ascendente con respecto al valor del s_i.

Example

Input	Output
6 3 6 5 1 4 2 3 7 8 6	7 3 6 2 8 2

Explanation

Para el caso de prueba del ejemplo se tienen los conjuntos $A = \{6,\ 5,\ 1,\ 4,\ 2,\ 3\}$ y $S = \{7,\ 8,\ 6\}$. Para los cuales se tienen tres subconjuntos de tamaño dos cuya suma es igual a $s_1 = 7$, estos son $\{1,\ 6\}$, $\{2,\ 5\}$ y $\{3,\ 4\}$. Se tienen dos subconjuntos de tamaño dos cuya suma es igual a $s_2 = 8$, estos son $\{2,\ 6\}$ y $\{3,\ 5\}$. Por último, se tienen dos subconjuntos de tamaño dos cuya suma es igual a $s_3 = 6$, estos son $\{1,\ 5\}$ y $\{2,\ 4\}$.

Use fast I/O methods

Problem 15. How Many Subsets?

Source file name:	Subsets.c, Subsets.cpp, Subsets.java, Subsets.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`ICPC Bolivia 2019 - Contest 10 RPC 2019`

We have a set A of positive integers with cardinality N ($|A| = N$, remember that the cardinality of a set is the total of different elements it has). You want to find out how many subsets of size two have a sum of their elements less than or equal to S.

For example, if you have the following set $A = \{6,\ 5,\ 1,\ 4,\ 2,\ 3\}$ and $S = 7$, then there is a total of 9 subsets of size two whose sum of its elements is less than or equal to 7, said subsets are: $\{6,\ 1\}$, $\{5,\ 1\}$, $\{5,\ 2\}$, $\{1,\ 4\}$, $\{1,\ 2\}$, $\{1,\ 3\}$, $\{4,\ 2\}$, $\{4,\ 3\}$ and $\{2,\ 3\}$.

Your mission, if you decide to accept it, is to count the total of subsets of size two that have a sum of their elements less than or equal to S.

Input

The input of the problem consists of a single test case. The test case contains three lines, the first line contains two positive integer numbers n ($1 \leq n \leq 10^6$) and q ($1 \leq q \leq 10^2$), which represent respectively the cardinality of the set A and the total of queries that will be made on the set A. The next line contains exactly n space-separated positive integer numbers A_1, A_2, A_3, ..., A_n ($1 \leq A_i \leq 10^8$, for $1 \leq i \leq n$), it is obviously guaranteed that the n elements of the set A are different. The next line contains exactly q space-separated positive integer numbers S_1, S_2, S_3, ..., S_q ($1 \leq S_j \leq 2 \times 10^8$, for $1 \leq j \leq q$), for the queries.

Output

Your program should print q lines, each of them containing a single value that represents the total result of subsets of size two that the sum of their elements is less or equal to S.

Example

Input	Output
6 3 6 5 1 4 2 3 7 8 12	9 11 15

Use fast I/O methods

Problem 16. Attractive Subsequence

Source file name:	Attractive.c, Attractive.cpp, Attractive.java, Attractive.py
Input:	Standard
Output:	Standard
Author(s):	Hugo Humberto Morales Peña - UTP Colombia
Source:	Maratón de Programación UFPS 2018 - Contest 06 RPC 2018

You receive a sequence S of non-negative integer numbers. Your task is to calculate the total of **Attractive Subsequences**. An **Attractive Subsequences** is a subsequence of consecutive elements in S that the sum of the elements in it is equal to the value K. For example, consider the sequence $S = \langle 0,\ 0,\ 25,\ 0,\ 0,\ 25\rangle$ and the value $K = 25$, there are 12 Attractive Subsequences, these are represented with ordered pairs $(ind_1,\ ind_2)$, ind_1 is the position of the first element and ind_2 is the position of the last element in the original sequence S. In this representation the Attractive Subsequences are: $(1,\ 3)$, $(1,\ 4)$, $(1,\ 5)$, $(2,\ 3)$, $(2,\ 4)$, $(2,\ 5)$, $(3,\ 3)$, $(3,\ 4)$, $(3,\ 5)$, $(4,\ 6)$, $(5,\ 6)$ y $(6,\ 6)$.

Input

The first line in the input contains one integer number T ($1 \leq T \leq 20$), the total of test cases in the input. Each test case contains three lines, the first line contains two positive integer numbers N ($1 \leq N \leq 10^5$) and Q ($1 \leq Q \leq 10^3$), the number of elements in the sequence S and the number of queries respectively. The next line contains exactly N space-separated non-negative integer numbers S_1, S_2, S_3, ..., S_N ($0 \leq S_i \leq 10^3$, for $1 \leq i \leq N$), that is the sequence S. The next line contains exactly Q space-separated positive integer numbers K_1, K_2, K_3, ..., K_Q ($1 \leq K_j \leq 10^7$, for $1 \leq j \leq N$), for the queries of the attractive subsequences.

Output

For each case you must print Q space-separated integer numbers in a single line, one per query, with the total of the attractive subsequences.

Example

Input	Output
2 6 2 0 0 25 0 0 25 25 50 7 3 1 6 5 2 3 4 7 7 5 11	12 3 4 2 2

Use fast I/O methods

Problem 17. DPA Numbers II

Source file name:	DPA02.c, DPA02.cpp, DPA02.java, DPA02.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2017 - Contest 03 RPC 2017`

In number theory, a positive integer belongs to one and only one of the following categories: Deficient, Perfect or Abundant (DPA).

To decide the category of a positive integer n, first you have to calculate the sum of all its proper positive divisors. If the result is less than n then n is a deficient number, if the result is equal to n then n is a perfect number and if the result is greater than n then n is an abundant number. Remember that the proper divisors of n don't include n itself.

For example, the proper divisors of the number 8 are 1, 2 and 4 which sum 7. Since $7 < 8$ therefore 8 is a deficient number. The proper divisors of the number 6 are 1, 2 and 3 which sum 6. Since $6 = 6$ therefore 6 is a perfect number. The proper divisors of the number 18 are 1, 2, 3, 6 and 9 which sum 21. Since $21 > 18$ therefore 18 is an abundant number.

The task is to choose the category of a positive integer n as a deficient, perfect or abundant number.

Input

Input begins with an integer t ($1 \leq t \leq 1100$), the number of test cases, followed by t lines, each line containing an integer n ($2 \leq n \leq 10^{12}$).

Output

For each test case, you should print a single line containing the word `deficient`, `perfect` or `abundant` that representing the category of the number n.

Example

Input	Output
10	deficient
5	perfect
6	deficient
16	abundant
18	deficient
21	perfect
28	deficient
29	abundant
30	abundant
40	deficient
43	

Use fast I/O methods

Problem 18. Rotations

Source file name:	Rotations.c, Rotations.cpp, Rotations.java, Rotations.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2017 - Contest 03 RPC 2017`

Humbertov Moralov in his student days, he is attending System Engineering at "University of Missing Hill" in *The Heaven's Branch Office* (Colombia, South America). He has attended the course of Assembly Language (at the first half of 1997).

The course of Assembler was fantastic!, with topics very interesting that the bit manipulation, where was worked right shifts, left shifts, rotations, masks and another bitwise operations (and, or, xor, not). And the best, in that course was worked interesting programming challenges. One of those programming challenges is the that proposes us today.

The programming challenge selected has name of "Rotations" and consists in the follow when that worked whit unsigned integer of eight bits (a byte), decide if the number n generate all the eight numbers from 0 at 7 with packages of consecutive three-bits. For to work on this problem you can use rotations, where the least significant bit (located on the far right of the number) can rotate and occupy the place of the most significant bit (located on the far left of the number).

For example, the number 226 has the binary representation 11100010 ($b_7b_6b_5b_4b_3b_2b_1b_0$), the eight packages of consecutive three-bits that are generated are the followings:

- $b_2b_1b_0 = (010)_2 = 2$
- $b_3b_2b_1 = (001)_2 = 1$
- $b_4b_3b_2 = (000)_2 = 0$
- $b_5b_4b_3 = (100)_2 = 4$
- $b_6b_5b_4 = (110)_2 = 6$
- $b_7b_6b_5 = (111)_2 = 7$
- $b_0b_7b_6 = (011)_2 = 3$
- $b_1b_0b_7 = (101)_2 = 5$

20 years have passed, Today computers are more powerful and faster, for this reason the professor *Humbertov Moralov* wants that you to work the programming challenge of the "Rotations" for unsigned integers of 32 bits (four bytes), you must validate if the number n generate all the 32 numbers from 0 at 31 with packages of consecutive five-bits.

Input

Input begins with an integer t ($1 \leq t \leq 3 \times 10^5$), the number of test cases, followed by t lines, each line contains an integer n ($0 \leq n \leq 4 \times 10^9$).

Output

For each test case, you should print a single line containing the word `yes` or `no` depending if the integer number n produce or not produce all the numbers from 0 to 31 with packages of consecutive five-bits.

Example

Input	Output
15	no
65535	no
65259922	yes
81354525	no
112805325	no
122525196	yes
192052550	yes
225525450	no
299525510	yes
318353525	no
344152934	yes
502445252	yes
522595252	no
1296752550	yes
3999995011	no
4000000000	

Use fast I/O methods

Problem 19. Ulam' Sequence

Source file name:	Ulam.c, Ulam.cpp, Ulam.java, Ulam.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`Coding Cup 2017 - ITSUR México`

An Ulam number is a member of an integer sequence devised by and named after *Stanislaw Ulam*, who introduced it in 1964. The standard Ulam sequence (the (1, 2)-Ulam sequence) starts with $U_1 = 1$ and $U_2 = 2$. Then for $n > 2$, U_n is defined to be the smallest integer that is the sum of two distinct earlier terms in exactly one way and larger than all earlier terms.

As a consequence of the definition, 3 is an Ulam number $(1 + 2)$, and 4 is an Ulam number $(1 + 3)$, here $2 + 2$ is not a second representation of 4, because the previous terms must be distinct. The integer 5 is not an Ulam number, because $5 = 1 + 4 = 2 + 3$.

For this problem, you have to write a program that calculates the n-th term in Ulam's sequence. That is, determine U_n.

Input

Input begins with an integer t $(1 \leq t \leq 4 \times 10^3)$, the number of test cases, followed by t lines, each line contains an integer n $(1 \leq n \leq 10^4)$.

Output

For each test case, you should print a single line containing the value of U_n.

Example

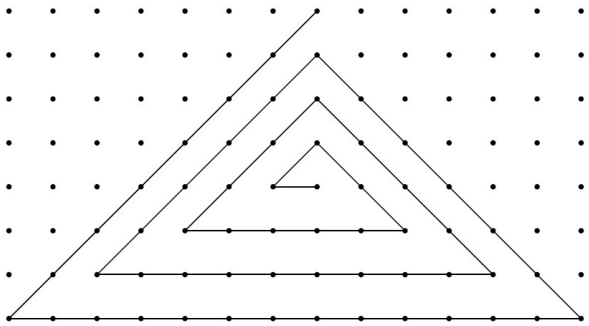
Input	Output
15 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	1 2 3 4 6 8 11 13 16 18 26 28 36 38 47

Use fast I/O methods

Problem 20. Humbertov and the Triangular Spiral

Source file name:	Spiral.c, Spiral.cpp, Spiral.java, Spiral.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2017 - Contest 03 RPC 2017`

Recently the professor *Humbertov Moralov* was sick, he had a fever and when he went to bed, he began to have a delirious dream. In the dream he draw over and over again a triangular spiral that began in the origin of the cartesian plane (coordinate $(0, 0)$) and the triangular spiral got bigger every time linking integer coordinates in the cartesian plane. For clarity, the triangular spiral is presented below:



The dream was so disturbing and it was repeated so many times, that when *Moralov* woke up, he remembered perfectly the triangular spiral, and for this reason he drew the previous graphic.

In the dream *Moralov* was disturbed and intrigued because he didn't know if all the integer coordinates could be reached at some point in the triangular spiral and, if that was the case, he also didn't know what would be the coordinate in the cartesian plane of the n-th point that is reached when drawing the triangular spiral. The first doubt was immediately resolved when the professor did the graphic ... all the points (integer coordinates) of the cartesian plane are eventually reached by the triangular spirals! Now the professor *Moralov* needs your help to indicate the coordinate in the cartesian plane of the n-th point that is reached when drawing the triangular spiral.

Input

Input begins with an integer t ($1 \leq t \leq 5 * 10^5$), the number of test cases, followed by t lines, each line contains an integer n ($1 \leq n \leq 10^{12}$).

Output

For each test case, you should print a single line containing two integers, separated by a space, denoting the coordinates x and y in the Cartesian coordinate system of point n in the triangular spiral.

Example

Input	Output
15	0 0
1	-1 0
2	0 1
3	1 0
4	2 -1
5	1 -1
6	0 -1
7	-1 -1
8	-2 -1
9	-3 -1
10	-2 0
11	-1 1
12	0 2
13	1 1
14	2 0
15	

Use fast I/O methods

Problem 21. How Many Digits does N have? I

Source file name:	Digits01.c, Digits01.cpp, Digits01.java, Digits01.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2018 - Contest 05 RPC 2018`

Recently, Professor *Humbertov Moralov* was organizing his workplace, he needed space to be able to locate his christmas decorations. For this reason, he began by selecting documents to either throw in the trash or to recycle. In this process he found some freehand notes with the puzzle: How many digits does $N = 2^{2003} \cdot 5^{2007}$ have?

Using this interesting puzzle as our starting point, the problem you are asked to solve now is: Given two nonnegative integers a, b ($0 \leq a,\ b \leq 10^5$), compute the total number of digits of $N = 2^a \cdot 5^b$.

Note: it is guaranteed that $|a - b| \leq 10^4$.

Input

Input begins with an integer t ($1 \leq t \leq 5 \times 10^5$), the number of test cases, followed by t lines, each one containing two nonnegative integers a, b ($0 \leq a,\ b \leq 10^5, \quad |a - b| \leq 10^4$).

Output

For each test case, you should print a single line containing one integer, denoting the total number of digits of N.

Example

Input	Output
8 10 8 8 10 2003 0 0 2007 2003 2007 1000 100 10000 10000 20000 10000	9 10 603 1403 2006 371 10001 13011

Use fast I/O methods

Problem 22. How Many Digits does N have? II

Source file name:	Digits02.c, Digits02.cpp, Digits02.java, Digits02.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2018 - Contest 05 RPC 2018`

Recently, Professor *Humbertov Moralov* was organizing his workplace, he needed space to be able to locate his christmas decorations. For this reason, he began by selecting documents to either throw in the trash or to recycle. In this process he found some freehand notes with the puzzle: How many digits does $N = 2^{2003} \cdot 5^{2007}$ have?

Using this interesting puzzle as our starting point, the problem you are asked to solve now is: Given two nonnegative integers a, b ($0 \leq a,\ b \leq 10^5$), compute the total number of occurrences of the digits 0, 1, 2, ..., 9. And finally the total number of digits of $N = 2^a \cdot 5^b$.

Note: it is guaranteed that $|a - b| \leq 5 \times 10^3$.

Input

Input begins with an integer t ($1 \leq t \leq 4 \times 10^5$), the number of test cases, followed by t lines, each one containing two nonnegative integers a, b ($0 \leq a,\ b \leq 10^5, \quad |a - b| \leq 5 \times 10^3$).

Output

For each test case, you should print out eleven single lines. The first 10 lines contain the number of occurrences of each digit. The eleventh line contains a positive integer with the total number of digits of N.

Example

Input	Output
4 10 8 8 10 2003 0 0 2007	8 0 0 0 1 0 0 0 0 0 9 8 0 1 0 0 1 0 0 0 0 10 68 71 62 70 57 58 66 58 46 47 603 145 148 131 145 137 128 124 149 140 156 1403

Use fast I/O methods

Problem 23. Asignación de citas para la vacuna del Covid 19

Source file name:	Covid19.c, Covid19.cpp, Covid19.java, Covid19.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Eddy Ramírez Jiménez - UNA & TEC Costa Rica`
Source:	`ICPC Centroamérica 2021 - Contest 08 RPC 2021`

Debido a la pandemia actual los centros de vacunación tienen que atender a una población enorme, por lo que han catalogado a las personas para poder atender a las personas con mayor riesgo primero. Pero algunas personas se han equivocado en cuándo debían de vacunarse, han fallado en identificar su verdadera prioridad y han atendido tarde a la cita de vacunación.

Sin embargo, la directriz del Ministerio de Salud es clara, cualquier persona que se haya inscrito para la vacunación, debe ser llamada acorde con su prioridad para ser vacunado. Además, como un dato no menor, nuestro país cuenta con un contrato de exclusividad con la *Compañía Johnson & Johnson* que fabrica la vacuna *Janssen*, la cual es de una sola dosis.

Si dos personas tuvieran la misma prioridad, entonces la primera en haber sido ingresada al sistema, iría primero.

Para ello debe hacer un programa que responda de la manera más eficiente posible sobre quién es el siguiente en ser llamado a vacunar.

Input

La entrada del problema consiste de un solo caso de prueba.

El caso de prueba contiene como máximo 10^6 líneas. Hay dos tipos de líneas, la de "ingreso" de la información de una persona al sistema y la de "llamado" para citar a una persona a que se acerque a un centro de vacunación a recibir la vacuna.

Una línea de "ingreso" contiene una cadena sin espacios (de máximo 20 letras en minúscula [a-z] del alfabeto en inglés), la cual representa el nombre de una persona y un número entero positivo p ($1 \leq p \leq 1000$) el cual representa la prioridad (entre más alto el valor, más alta es la prioridad).

Una línea de "llamado" contiene únicamente una cadena de longitud uno con la letra mayúscula `V` (de vacuna).

El final del caso de prueba es dado por el fin de archivo (End Of File - EOF).

Output

Para cada caso de prueba, su programa debe imprimir tantas líneas como líneas de "llamado" hay en la entrada, cada una de estas líneas debe contener una cadena con el nombre de la persona que debe ser vacunada. Si no hay personas registradas en el sistema en el momento del "llamado" entonces se debe dejar (es decir, se debe imprimir) una línea en blanco.

Example

Input	Output
laura 1 oscar 100 emilio 20 V V andres 30 V	oscar emilio andres
hugo 100 humberto 90 V V V rafael 80 V	hugo humberto rafael
alvaro 100 beatriz 100 carlos 100 diana 100 esperanza 100 fabio 100 V V V V V	alvaro beatriz carlos diana esperanza

Problem 24. Emergency Room

Source file name:	Emergency.c, Emergency.cpp, Emergency.java, Emergency.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`Maratón de Programación UFPS 2020 - Contest 13 RPC 2020`

In the past days, professor *Humbertov Moralov* suffered a small accident on his left foot and had to go to the hospital for "treatment" in the emergency room. During the almost four hours of waiting, the professor had time to analyze and understand how the Colombian emergency service works.

Patients requesting medical attention are constantly arriving at the emergency department, which are initially evaluated by a doctor based on a method called Triage, in which the priority of the emergency is determined. The possible Triage ratings are 1, 2, 3, 4 and 5, with 1 being the rating that indicates that care should be immediate (the patient's life is at high risk) and 5 being the rating that indicates that the patient needs medical attention but that no vital organs are involved and therefore can wait for five or six hours.

After the doctor's assessment, patients are taken to the waiting room and there they are called in order of priority with respect to the Triage rating. When there are several patients with the same priority based on Triage, the one who arrived first must be treated. If this is not respected, a "riot" is generated by the patients and the emergency department is destroyed!

Professor Humbertov Moralov needs help and asks you, as a student in an academic *Computer Science* program, to develop a computational solution to determine the order in which patients are treated in the emergency department.

Input

The problem input consists of several test cases. Each test case begins with a line containing a positive integer n ($3 \leq n \leq 2 \cdot 10^5$) corresponding to the total number of transactions made in the emergency department.

Then follow n lines, each of them begins with a pair of positive integers t ($t \in \{1, 2\}$) and h ($1 \leq h \leq 8 \cdot 10^5$) corresponding respectively to the type of transaction (1 indicates that a patient requests the emergency service and 2 indicates that a doctor attends the next emergency) and the time in which the transaction occurs (in seconds). It is guaranteed that $h_i > h_{i-1}$ for $2 \leq i \leq n$.

For type 1 transactions, the line is complemented by a digit p ($p \in \{1, 2, 3, 4, 5\}$) and a string s (it consists only of uppercase English letters ($A - Z$) and its length is ≤ 20) that correspond respectively to the rating based on Triage and to the patient's name.

The end of the test cases is given by the End Of File (EOF).

Output

For each test case, print as many lines as there are type 2 transactions. Each of these lines must contain three positive integers a, b, c ($1 \leq a, b, c \leq 8 \cdot 10^5$; $a < b$) and a string s representing respectively the time of arrival, the time of service, the total waiting time and the patient's name.

Example

Input	Output
10 1 1 5 CARLOS 1 3 4 MANUEL 1 4 4 ANDRES 1 6 1 GABRIEL 2 9 1 13 2 YEFRI 2 20 1 24 1 STEVEN 2 30 2 50	6 9 3 GABRIEL 13 20 7 YEFRI 24 30 6 STEVEN 3 50 47 MANUEL

Use fast I/O methods

Problem 25. Interval

Source file name:	Interval.c, Interval.cpp, Interval.java, Interval.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2018 - Contest 05 RPC 2018`

Let's take a list L containing n ($1 \leq n \leq 10^6$) positive integer numbers in the interval $[1, 10^5]$, and assume you have three operations to work on it: Query, Insert and Delete.

The format for each operation is: p y, p indicates the type of the operation, 1 = Query, 2 = Insert, and 3 = Delete. y represents the value where the operation will be applied.

The **Query** operation must print a pair of numbers x z, such that $x = \max(\{v \in L \mid v < y\})$, and $z = \min(\{v \in L \mid y < v\})$

The **Insert** operation adds y to the list.

The **Delete** operation removes one occurrence of the element y in L, if the element is present.

Note 1: For the **Query** and **Delete** the element y might not exist.

Nota 2: There can be several occurrences of y inside the list.

Input

Input begins with an integer t ($1 \leq t \leq 10$), the number of test cases. The first line of the test contains a positive integer n ($1 \leq n \leq 10^6$), with the lenght on the list L. The second line contains exactly n space-separated positive integer numbers L_1, L_2, L_3, ..., L_n ($1 \leq L_i \leq 10^5$, for $1 \leq i \leq n$), that is the list L. The test case continues with a line containing a number q ($1 \leq q \leq 10^5$), the number of operations to apply over the list. In each one of the next q lines two integers p y, ($p \in \{1,\ 2,\ 3\}$, $1 \leq y \leq 10^5$) are presented.

Output

For each query of the type 1, you should print one single line containing the two integers x z. If there is not such value x then $x = -1$, if there is not such value z then $z = 100001$.

Input	Output
2 5 4 4 4 4 4 8 1 4 1 3 1 5 2 1 2 8 1 4 3 4 1 3 5 10 1 8 3 5 7 1 5 3 5 2 6 1 5 3 8 2 12 1 8	-1 100001 -1 4 4 100001 1 8 1 4 3 8 3 6 6 10

Use fast I/O methods

Problem 26. Pseudorandom

Source file name:	Pseudo.c, Pseudo.cpp, Pseudo.java, Pseudo.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`UTP Open 2016 - Contest 03 RPC 2016`

The most common procedure used in computers to generate pseudorandom numbers is the *linear congruential generator*.

The *linear congruential generator* uses four integer numbers: the multiplier a, the increment c, the module m and the seed x_0, with $2 \leq a < m$, $0 \leq c < m$ and $0 \leq x_0 < m$. It generates a sequence of pseudorandom numbers $\{x_n\}$, with $0 \leq x_n < m$ for all n, by applying successively the following congruence:

$$x_{n+1} = (a \cdot x_n + c) \ mod \ m$$

For example, the sequence of numbers that are generated when the multiplier $a = 7$, the increment $c = 4$, the module $m = 9$, and the seed $x_0 = 3$, is: 3, 7, 8, 6, 1, 2, 0, 4, 5, 3, 7, 8, 6, ...

In this sequence, we notice that $x_0 = x_9 = 3$,$x_1 = x_{10} = 7$, $x_2 = x_{11} = 8$, and so on. That means that starting from some point the values start being generated again in a loop. This happens because in the linear congruential generator each term in the sequence depends only on the previous term.

The lineal congruence generator doesn't always produce m different terms, for example, when the multiplier $a = 3$, the increment $c = 4$, the module $m = 9$ and the seed $x_0 = 5$, the following sequence is generated: 5, 1, 7, 7, ..., we can see that there are three different elements in the sequence, but the first two (5 and 1) are only generated once and the third term is generated in a loop.

The task is: given the values of a, c, m and x_0, indicate if the linear congruential generator produces m distinct values before values start repeating. You should also print the total amount of elements that are generated before any repeated number appears, the total amount of elements that are generated only once, and the total amount of elements that are generated repeatedly in the loop.

Input

There are several test cases of input. Each test case consist of a single line that has four integer numbers: a, c, m, x_0 separated by spaces ($2 \leq a < m, \quad 0 \leq c < m, \quad 3 \leq m \leq 10^3, \quad 0 \leq x_0 < m$).

Output

For each test case you should print a single line containing the word `YES` or `NO` (indicating if m distinct elements are generated before the numbers start repeating), followed by the three integer numbers t s r, which indicate, respectively, the total amount of elements that are generated before any repeated number appears, the total amount of elements that are generated only once, and the total amount of elements that are generated repeatedly in the loop.

Example

Input	Output
7 4 9 3 3 4 9 5 2 3 9 4	YES 9 0 9 NO 3 2 1 NO 6 0 6

Use fast I/O methods

Problem 27. Generate, Sort and Search

Source file name:	Generateso.c, Generateso.cpp, Generateso.java, Generateso.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`
Source:	`ICPC Bolivia 2016 - Contest 10 RPC 2016`

We have the following recursive function:

$f(1) = x$
$f(n) = (a \cdot f(n-1) + c)\ mod\ m$, with $n \geq 2$, $n \in \mathbb{Z}^+$

Remember that the operation *mod* calculates the remainder of the integer division.

With the previous recursive function you should generate a sequence containing the first n elements, which are: $f(1)$, $f(2)$, $f(3)$, $f(4)$, ..., $f(n)$. Then, you should sort those numbers in ascending order (with respect to its value), so you can tell which number is located in the ith position of the sorted sequence.

Input

There are several test cases. The first line of each test case has six integer numbers: a, c, m, x, q, n separated by spaces ($2 \leq a < m$, $0 \leq c < m$, $3 \leq m \leq 10^3$, $0 \leq x < m$, $1 \leq q \leq 10^4$, $1 \leq n \leq 10^8$). The remaining lines of each test case have q integer numbers. Each one corresponds to the position in the sorted sequence whose value wants to be known.

Output

For each query you should print a single line containing the integer number in the ith position of the sorted sequence.

Example

Input	Output
7 4 9 3 5 10 2 10 3 9 4	1 8 2 7 3

Use fast I/O methods